Dancing with Still Images: Video Distillation via Static-Dynamic Disentanglement

Ziyu Wang* Yue Xu* Cewu Lu Yong-Lu Li†
Shanghai Jiao Tong University
{wangxiaoyi2021, silicxuyue, lucewu, yonglu_li}@sjtu.edu.cn

Abstract

Recently, dataset distillation has paved the way towards efficient machine learning, especially for image datasets. However, the distillation for videos, characterized by an exclusive temporal dimension, remains an underexplored domain. In this work, we provide the first systematic study of video distillation and introduce a taxonomy to categorize temporal compression. Our investigation reveals that the temporal information is usually not well learned during distillation, and the temporal dimension of synthetic data contributes little. The observations motivate our unified framework of disentangling the dynamic and static information in the videos. It first distills the videos into still images as static memory and then compensates the dynamic and motion information with a learnable dynamic memory block. Our method achieves state-of-the-art on video datasets at different scales, with a notably smaller memory storage budget. ***Our code is available at https://github.com/yuz1wan/video_distillation****.*

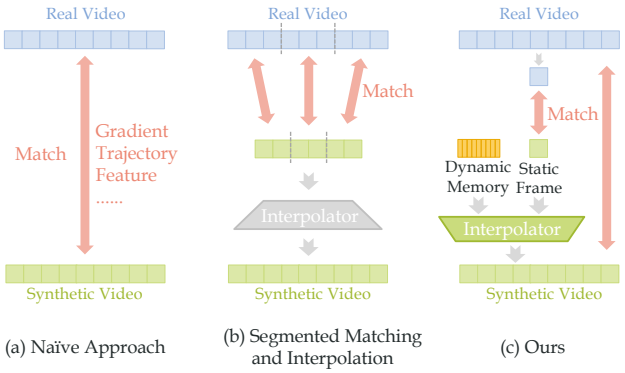


Figure 1. (a) Naive video distillation methods simply match the training dynamics (gradient, feature, trajectory, *etc*.) of the real and synthetic videos. (b) To exploit the temporal redundancy of videos, we propose a paradigm with segmented matching and interpolation techniques to cover all levels of temporal condensation. (c) Based on this paradigm and our observations, we propose an approach of efficient static frame distillation and motion compensation, with better efficiency and performance.

1. Introduction

Dataset distillation, as an emerging direction recently, compresses the original dataset into a smaller one while maintaining training effectiveness. It alleviates the challenges of costly training due to the increasingly large datasets and models. It is widely adopted in various downstream fields including federated learning and continual learning.

Recent works on dataset distillation mainly focus on distilling images [4, 6, 7, 10, 13, 15–17, 23, 25, 29–32]. Though some methods seem to seamlessly adapt to other data formats or modalities [12, 27, 28], few works studied video distillation. Compared to image data, videos possess an additional temporal dimension, which significantly adds to the time and space complexity of the distillation algorithms and is already hardly affordable when the instance-per-class (IPC) is large. Besides, the scale of video datasets [3, 11, 19] is usually more intimidating. However, the high temporal redundancy of videos is very suitable for and can be well exploited by dataset distillation methods, providing a good opportunity for dataset distillation. Therefore, in this work, we firstly and systematically study the dataset distillation on video data, especially involving the compression of the video temporal redundancy.

Currently, dataset distillation approaches directly align the training dynamics (gradient [31], trajectory [4], feature [30], *etc*.) of real and synthetic. On video datasets, these methods simply match all the real and synthetic frames and the frame correspondence can be depicted as a complete bipartite graph. Therefore, to condense the temporal dimension, we can either reduce the synthetic frames or prune the bipartite correspondence graph between real and synthetic frames. For the further analysis of temporal condensation in video distillation, we put these two schemes under one unified framework, namely "segmented

*The first two authors contribute equally.
†Corresponding author.

matching and interpolation" (Fig. 1(b)): the real videos are synthetic videos are partitioned to multiple segments and distillation are applied within each real-synthetic segments pair; the synthetic videos are then interpolated to the required video length. This framework can cover most scenarios at different levels of temporal condensation.

Then, to thoroughly study temporal condensation, we build a taxonomy for various temporal correspondences and classify the potential methods based on four essential dimensions: the numbers of synthetic frames and real frames for matching, the number of segments, and the interpolation algorithm. Along these four dimensions, we conduct comparisons of the distillation performance and computation cost. Empirical analysis shows that, though increasing frame numbers does enhance distillation performance, the improvement is marginal and comes at the expense of considerably longer training times and higher costs; and frame segmentation reduces the training costs, but brings a substantial decrease in model performance. These important taxonomies and observations could guide the research and the efficient algorithm design of video distillation.

In light of the above observation, we propose a unified framework for video distillation to exploit the unique temporal redundancy of videos. Our observation implies that dense frame correspondence is non-critical in the video distillation task. Hence, to maximize the efficiency, we reduce the real frames, synthetic frames, and segment length to 1. This is equivalent to image distillation with which we can distill the *static information* in the first stage. Second, we use a learnable *dynamic memory* to compensate for the loss of dynamic information. The static and dynamic memories are then combined with an integrator network. Our paradigm can be effortlessly applied to existing algorithms to enhance performance with a memory storage budget (referred to as *storage* here) used.We embed our method to various data distillation algorithms including DM [30], MTT [4], FRePo [32], and achieve state-of-the-art with less storage. Our approach could achieve comparable performance with $<50\%$ memory storage budget and bring substantial improvement with a comparable one.

Overall, our contributions are: (1) We propose the very first work that systematically studies the video dataset distillation. (2) We introduce a novel taxonomy for temporal condensation in video distillation methods, which guides our and future works. (3) We propose a novel paradigm, enabling existing image distillation techniques to achieve improved results when applied to video distillation while using an even smaller memory storage budget.

2. Related work

Dataset Distillation/Condensation. Dataset distillation [25], endeavors to condense large datasets into smaller ones while maintaining comparable training performance. The algorithms fall into the following categories:

(1) *Performance Matching*: following the very first work of DD [25], a broad category of techniques employs bi-level optimization. A few methods integrate kernel ridge regression (KRR) to reduce the computational complexity of bi-level optimization, where KIP [16, 17] employs the Neural Tangents Kernel (NTK), while RFAD [15] adopts the Empirical Neural Network Gaussian Process (NNGP). FRePo [32] separates a neural network into a feature extractor and a linear classifier to optimize.

(2) *Parameter Matching*: DC [31] aligns the single-step gradients of synthetic and real data. In line with DC, DSA [29] enhances this approach through symmetrical image augmentation, and IDC [10] enhances by storing synthetic data in lower resolutions. MTT [4] first applies multi-step parameter matching, and TESLA [6] reduces memory usage and uses learnable soft labels.

(3) *Distribution Matching*: DM [30] directly aligns the features of real and synthetic data, while CAFE [23] ensures statistical feature properties from synthetic and real samples are consistent across all network layers except the final one.

(4) *Factorization Methods* decompress the full dataset into two components: base and hallucinator. HaBa [13] employs task-input-like bases and ConvNet hallucinators, while LinBa [7] integrates a linear hallucinator with given predictands. Inspired by these methods, we factorize the static and dynamic information in video distillation to minimize temporal redundancy and reduce storage costs.

Video Recognition. Video recognition involves the classification of videos into semantic classes, *e.g.*, human actions and scenes. Currently, there are several main design philosophies for video recognition with deep learning: (1) *2D Convolution-Based*: The most intuitive approach is to break down the video into individual frames and process the frames individually. Then temporal aggregation (pooling, LSTM, GRU, *etc.*) is used for getting video-level features, which is then used for classification. (2) *3D Convolution-Based*. To adopt early aggregation of temporal features, the presence of an additional temporal dimension in videos naturally suggests the possibility of employing 3D convolutional networks. Tran et al. [21] propose C3D, and then Carreira and Zisserman [3] extend the pre-trained models of 2D convolutional networks to 3D, with consideration of alleviating the challenges posed by the large number of parameters in 3D convolutions. Meanwhile, efforts are dedicated to low-rank approximations for 3D convolutions. R(2+1)D [22] employs a spatial 2D convolutional structure combined with a temporal 1D convolution to achieve pseudo-3D convolutions. (3) *Transformer-Based*. With the success of attention mechanisms in natural language processing, the long-range effectiveness of self-attention determines its suitability for video recognition. Therefore, there have also emerged some mod-

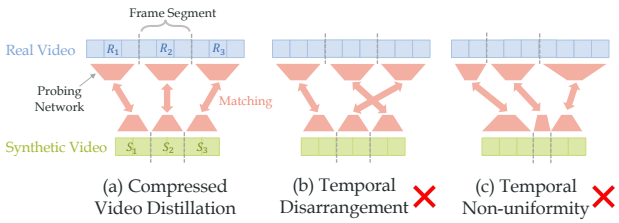


Figure 2. The distillation setting in (a) obeys temporal consistency, while (b) and (c) violate the two consistency preconditions.

els [2, 20, 24] for videos based on Transformers.

3. Pre-analysis

The temporal redundancy has been widely discussed for video understanding, while we focus on analyzing the temporal compression in dataset distillation. In this section, we first propose some basic principles for temporal compression (Sec. 3.1). We use a generic paradigm to describe the temporal compression strategies and further propose a taxonomy of compression according to four factors (Sec. 3.2), along with which we conduct comprehensively comparisons and obtain our observations and conclusions, supporting further study on video distillation (Sec. 3.3).

3.1. Preliminaries

Video Temporal Redundancy. Most current works are dedicated to developing and improving methods for compressing image datasets in terms of quantity. Video data have an additional temporal dimension as its major difference from images, suggesting the presence of temporal redundancy. Video temporal redundancy study has a long history [9, 14]. Researchers have long observed the significant temporal redundancy, which has diverse causes. For example, videos inherently exhibit substantial similarity between adjacent frames, leading to low temporal information utilization during data usage. Thus, we focus on, analyze, and exploit the temporal compression for video distillation.

Precondition for Temporal Correspondence. To study the temporal compression, we are categorizing the temporal correspondence for the distillation matching (*e.g.* gradient/trajectory/distribution matching) between real and synthetic videos. We can put all scenarios into one formulation, which covers all possible methods from the temporal aspect:

Definition 1. Compressed video distillation involves real video R and synthetic video S with asymmetrical lengths. Multiple frame sequences are drawn from R, S and paired: (R_1, S_1), (R_2, S_2), $\cdots$, (R_K, S_K), where $R_i \subseteq R$, $S_i \subseteq S$, $\forall i = 1, \cdots, K$. Distillation is to apply matching algorithms to each frame sequence pair individually.

Fig. 2(a) gives an example of compressed video distillation. Though the existing distillation methods usually produce irregular and "freeform" patterns that are cryptic for humans, on video data, we intuitively desire an algorithm that obeys **temporal consistency**. Otherwise, the video algorithm could degrade due to the loss of correct temporal information. The temporal consistency is ensured by two preconditions: the real and synthetic frames sampled for distillation should be in the correct order and follow a uniform flow of time. More specifically, the orderedness is:

Precondition 1. (Orderedness) During the compressed video distillation of real video R and synthetic video S, given two frame sequence pairs (R_1, S_1), (R_2, S_2), we define a partial order $\leq_T$: $(R_1, S_1) \leq (R_2, S_2)$ *iff* any frame in R_1 occurs earlier than any frame in R_2 and any frame in S_1 occurs earlier than any frame in S_2. The distillation process obeys **orderedness** *iff* all the frame sequence pairs for R and S yield a total ordering associated with $\leq_T$.

That is, all frame sequence pairs are comparable and associated with $\leq_T$ and both real and synthetic sequences are in the correct time order. The matching strategy in Fig. 2(b) does not meet orderedness as the two matching pairs at right are in the wrong temporal order.

Precondition 2. (Uniformity) Compressed video distillation of real video R and synthetic video S, obeys **uniformity** *iff* $|R_1| = |R_2| = \cdots = |R_K|$, $|S_1| = \cdots = |S_K|$.

All the real frame sequences have the same length, as the same to synthetic frame sequences. This ensures that the synthetic video we learn follows a uniform flow of time, *e.g.*, the matching strategy in Fig. 2(c) leads to non-uniform FPS, and the synthetic frames at left will learn a much smaller FPS than the frames at right. We also justify the uniformity with experiments in the supplementary.

These two preconditions narrow the searching space of temporal condensation strategies, enabling us to systematically categorize and analyze the compressed video distillation. In Fig. 3(a), the naive algorithm may match all synthetic frames to real frames. To compress the temporal dimension, we can either reduce the number of frames (Fig. 3(b)) or reduce the correspondence by temporal segmentation (Fig. 3(c)). Note that we do not force orderedness and uniformity *within* each frame sequence pair for matching, since the distillation matching algorithms themselves could drive the synthetic data to follow the consistency.

3.2. Segmented Matching and Interpolation

To take a step further, we propose **Segmented Matching and Interpolation** framework for the quantitative comparison and analysis of temporal condensation (Fig. 3(e)). Given a target synthetic frame number L_{syn}, to approach flexible compression rate, we distill N_{real} real frames to N_{syn} frames and interpolate the frames to our target L_{syn}. Specifically, the real video and synthetic video are *segmented* evenly for the pairwise distillation matching, which

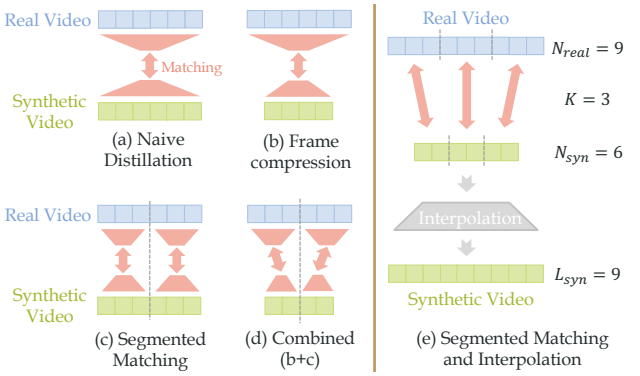


Figure 3. **Left**: Different types of video distillation that obey temporal consistency. **Right**: A basic framework for compressed video distillation, which condenses the temporal dimension by distillation, and interpolates the synthetic frames to the target length.

is the only valid strategy for temporal consistency. The interpolation enables the compression of synthetic video. This paradigm covers most video distillation methods from the perspective of temporal compression, and the extent of compression can be parameterized by the four factors:

(1) Number of Independent Synthetic Frames (N_{syn}) is the size of the trainable synthetic frames. These frames are learned with dataset distillation algorithms and will be interpolated to the target length of synthetic video L_s. Smaller N_{syn} indicates a larger temporal compression rate.

(2) Number of Real Frames (N_{real}) is the size of real frames for distillation matching. Larger N_{real} implies a larger receptive field for synthetic video during distillation.

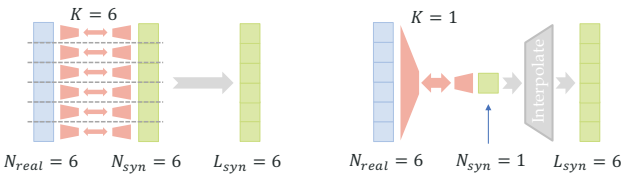
(3) Number of Segments (K). We cut the real and synthetic videos into the same number of segments and apply distillation between the pairs of real and synthetic segments. Larger K could reduce the training time since the distillation algorithm receives smaller segments with fewer frames.

(4) Interpolation Algorithm ($\mathcal{I}$) interpolates the N_{syn} independent synthetic frames to the required synthetic video length. Our algorithms are detailed in Sec. 3.3.

So the level of temporal compression can be uniquely determined by a quadruplet $(N_{syn}, N_{real}, K, \mathcal{I})$. We show some examples in Fig. 4 with different combinations. In the following section, we will compare and analyze these four axes and put forward some empirical conclusions to drive future studies on video distillation.

3.3. Comparison of Temporal Compression

To investigate the effects of different temporal compression levels, we conduct comprehensive experiments with different N_{syn}, N_{real}, K, and $\mathcal{I}$. In the following experiments, the DM [30] algorithm is adopted and we use ConvNet (3-layer convolutional network from [31]) with one layer GRU head [5] as our backbone network.



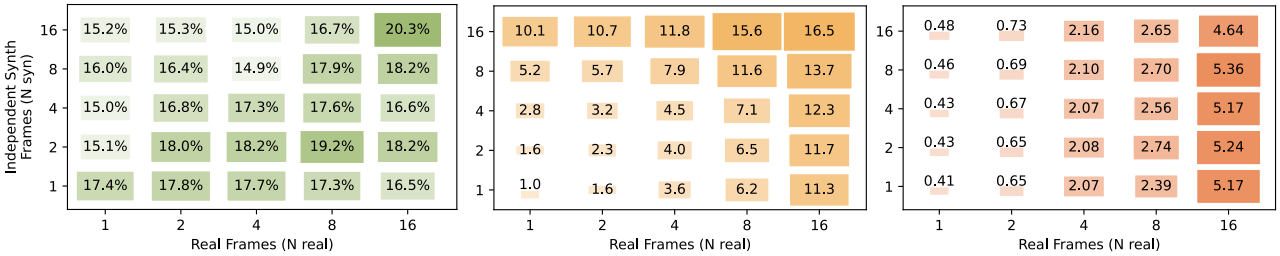
(a) Frame-wise Distillation (b) Distill video to still image

Figure 4. Examples with different compression levels. (a) use an image distillation algorithm to distill the frames one by one. (b) distills the video into a single image.

Comparison of N_{syn} and N_{real} values. We first compare the models with different N_{syn} (number of independent synthetic frames) and N_{real} (number of real frames) and visualize in Fig. 5. We implement the segmented matching segment number $K = 1$. The performance is Fig. 5(a) shows that: (1) distilling the video to still image could yield decent accuracy (over 17%), (2) larger N_{real} and N_{syn} brings *minor* performance gain (less than 3%), (3) the model degrades when $N_{real} < N_{syn}$ (left-top of the figure), potentially due to insufficient temporal information in the real data. Fig. 5(b) and (c) show that larger N_{real} and N_{syn} lead to significant memory consumption, *e.g.* model with $N_{real} = N_{syn} = 16$ takes $16\times$ GPU memory than a $N_{real} = N_{syn} = 1$ model. And models with large N_{real} also take more training time. And we can also read Fig. 5 diagonally to fix the per-frame receptive field N_{real}/N_{syn}, and basically larger ratio leads to better performance.

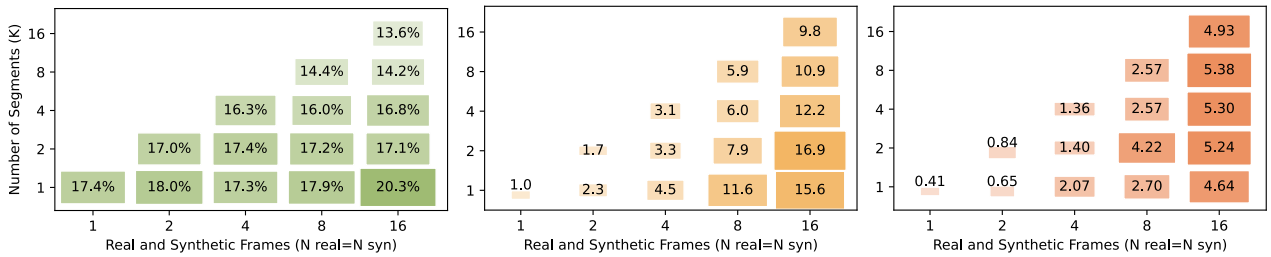
Comparison of K values (segments). We study the effects of K in Fig. 6. The segmentation could notably reduce memory consumption (up to 40%) while maintaining the training speed. However, the efficiency is achieved at the cost of performance as the segmentation decreases the "receptive field" of each synthetic frame.

Comparison of Interpolation Algorithms $\mathcal{I}$. $\mathcal{I}$ is critical to compressed video distillation, especially when N_{syn} is small. We use various interpolation methods: **(1) Duplication** is simply copying the learned synthetic frames to the required length, or namely *nearest interpolation*. *e.g.* a 2-frame video $[f_1, f_2]$ can be interpolated to 4-frame video $[f_1, f_1, f_2, f_2]$. **(2) Linear interpolation** generates intermediate frames by blending adjacent reference frames. The frame f_t at time t will be the weighted sum of nearest reference frames f_{t_1}, f_{t_2} according to their temporal distance $t_2 - t$ and $t - t_1$. *e.g.* a 2-frame video $[f_1, f_2]$ can be interpolated to 4-frame video $\left[f_1, \frac{2f_1+f_2}{3}, \frac{f_1+2f_2}{3}, f_2\right]$. **(3) Parametric interpolator** is a pre-trained interpolation network on the real video dataset. For each video data with L_{syn} frames, we evenly sample N_{syn} frames and duplicate them to length L_{syn}. We train a TimeSformer [2] model φ on these real data and it learns to recover the original video from the duplicated one. The pretrained model φ can be utilized for interpolation, *e.g.* a 2-frame video $[f_1, f_2]$ can



(a) Distillation accuracy. (b) Memory consumption (GB). (c) Training time (seconds per iteration).

Figure 5. Model performance (a) and efficiency (b, c) comparison with different independent synthetic frames and real frames number N_{syn}, N_{real}, with DM [30] and ConvNet+GRU.



(a) Distillation accuracy. (b) Memory consumption (GB). (c) Training time (seconds per iteration).

Figure 6. Performance (a) and efficiency (b, c) comparison with different numbers of segments K, with DM [30] and ConvNet+GRU.

N_{syn} and N_{real}	1	2	4	8
Duplication	17.4 ± 0.3	18.0 ± 0.4	17.3 ± 0.5	17.9 ± 0.6
Linear	-	15.6 ± 0.1	16.1 ± 0.4	16.3 ± 0.5
Parametric	17.0 ± 0.6	18.6 ± 0.8	19.2 ± 1.0	18.5 ± 0.9

Table 1. Comparison of different interpolation algorithm $\mathcal{I}$ with DM [30] and ConvNet+GRU model.

be interpolated to 4-frame video $\varphi([f_1, f_1, f_2, f_2])$.

We compare the three interpolators in Tab. 1. The simple duplication method outperforms linear interpolation. The parametric method performs the best, especially on larger frame numbers, as it encodes dataset-specific inductive bias to compensate for the loss of temporal information.

Discussion. With the comparison of the four dimensions of temporal compression, we have some fundamental observations that could offer direction for our further study of video distillation: (1) Temporal compression confers significant advantages to dataset distillation and static images could encode more than little knowledge for video datasets; (2) Segmented distillation could reduce the distillation cost, but significantly sacrifice the model performance (3) Parametric interpolation could compensate for the loss of temporal dynamic information in the video.

4. Methodology

Dataset distillation is a lossy data compression process. With synthetic data as an intermediary, only part of the information in a real dataset could finally be learned by the model according to the data processing inequality. Thus,

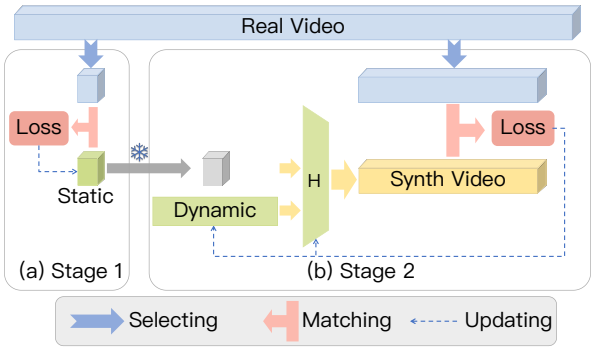


Figure 7. Our two-stage method: Stage 1: static memory learning with image distillation on one frame per video. Stage 2: the static (frozen) and dynamic memory are combined into synthetic videos by $\mathcal{H}$, and aligned with the real data.

based on the analysis in Sec. 3.3 and considering the trade-off between efficiency and efficacy, we propose a video dataset distillation paradigm by disentangling the static and dynamic information in videos. We put more effort into the learning of static information with low cost (Sec. 4.1), and then compensate for the dynamic information (Sec. 4.2). We give an overview of our method in Fig. 7.

4.1. Static Learning

The results in Sec. 3.3 indicate that static information in videos is more critical to the distillation task, given the limited capacity of small synthetic data. Hence, we use a $N_{syn} = N_{real} = K = 1$ setting to learn a **static mem-**

Algorithm 1 Static Learning and Dynamic Fine-tuning.

Input: Distillation matching loss $\mathcal{A}$, origin real video dataset $\mathcal{T}$, selection method $\mathcal{B}$.
Output: Static and dynamic memory $\mathcal{S}$, $\mathcal{D}$, network $\mathcal{H}$

Stage 1: Static Learning
Initialize $\mathcal{S}$ with random frames $\mathcal{T}$.
for $i = 1$ to M **do**
  $\mathcal{T}_s = \mathcal{B}(\mathcal{T}, 1)$ // Form a single-frame dataset
  $\mathcal{S} \leftarrow \mathcal{S} - \alpha_S \nabla_S \mathcal{A}(\mathcal{S}, \mathcal{T}_s)$ // Update $\mathcal{S}$
end for
Stage 2: Dynamic Fine-tuning
Initialize $\mathcal{D}$ and $\mathcal{H}$ with noise
for $i = 1$ to N **do**
  $\mathcal{T}_d = \mathcal{B}(\mathcal{T}, N_{real})$ // Form a multi-frame dataset
  $\mathcal{L} = \mathcal{A}(\mathcal{H}(\mathcal{D}, \mathcal{S}), \mathcal{T}_d)$ // Combine back to multi-frames and compute matching loss
  $\mathcal{H} \leftarrow \mathcal{H} - \alpha_H \nabla_H \mathcal{L}$ // Update $\mathcal{H}$
  $\mathcal{D} \leftarrow \mathcal{D} - \alpha_D \nabla_D \mathcal{L}$ // Update $\mathcal{D}$
end for

ory with only one frame. The specific distillation process involves selecting one frame randomly from each video segment to form an image dataset in each epoch. The DC [31] method is then applied for gradient matching on a convolutional network. Since a new image dataset is created in each epoch, the "image" we distill has ideally observed all video frames and distilled the static memory from them.

4.2. Dynamic Fine-tuning

The choice of dynamic memory can be diverse. In this paper, our dynamic memory is represented as multiple frames of single-channel images. We use a network $\mathcal{H}$ which takes static memory and dynamic memory as input and outputs video clips. At this stage, we fix the static memory and use different matching methods (performance matching, distribution matching, and parameter matching) to simultaneously update the network $\mathcal{H}$ and dynamic memory.

We use a concrete formula to explain our paradigm. We refer $\mathcal{A}(\mathcal{T}_{syn}, \mathcal{T})$ as a matching loss of distillation where $\mathcal{T}_{syn}$ is synthetic dataset and $\mathcal{T}$ indicates origin dataset $\{(x_i, y_i)\}_{i=1}^{|\mathcal{T}|}$, $x_i \in \mathbb{R}^{f_i \times c \times h \times w}$, $y_i \in \{0, 1, \ldots, C-1\}$. Given a frame selection method $\mathcal{B}(\mathcal{T}, N)$ selecting N frames from $\mathcal{T}$ to obtain a dataset with $x_i \in \mathbb{R}^{N \times c \times h \times w}$, we summarize our paradigm in Alg. 1.

5. Experiments

5.1. Datasets and Metrics

We adopt small video datasets UCF101 [19] and HMDB51 [11], and large-scale Kinetics [3] and Something-Something V2 [8] in our experiments. UCF101 [19] consists of 13,320 video clips in 101 action categories while HMDB51 [11] consists of 6849 video clips in 51 action categories. Kinetics [3] is a collection of video clips that cover 400/600/700 human action classes while SSv2 [8] covers 174 motion-heavy classes. To evaluate the distillation algorithms on more diversified data scales, and considering the efficiency of experiments and the clarity of model comparisons, following the scale of pioneering work on image distillation [25], we build a miniaturized version of UCF101, named **MiniUCF**, including 50 most common classes from the UCF101 dataset. This miniaturization enables rapid iterations of our method and facilitates observing relatively significant changes in performance. We report the top-1 classification accuracy for MiniUCF and HMDB51, and the top-5 classification accuracy for Kinetics400 and SSv2.

5.2. Baselines

The baseline methods involve: (1) coreset selection methods (random selection, Herding [26] and K-Center [18]) following the implementation for image distillation in DC [31]. (2) direct adaptation of the common image distillation methods (DM [30], MTT [4], FRePo [32]) to the video distillation task. (3) image distillation method (DC [31]) with frame duplication for a "boring videos" proposed by us, namely "Static-DC".

5.3. Implementation Details

Data. For MiniUCF and HMDB51, the videos are sampled to 16 frames with sampling interval 4 dynamically, $i.e.$. the frames indices vary in different epochs. Following the setup in C3D [21], each of these frames is cropped and resized to 112x112. For Kinetics-400 and Something-Something V2, the videos are sampled to 8 frames before the distillation, and the frames are cropped to 64x64. We only use horizontal flipping with a 50% probability for data augment.

Static Learning. We use DC [31] to distill static memory with random real frame initialization. We utilize a 4-layer 2D convolutional neural network for distillation (ConvNetD4) and perform an early stop in the distillation training when the loss converges. Interestingly, our experiments indicate that static memory is not necessarily trained to full convergence, as dynamic memory compensates for it.

Dynamic Finetuning. In dynamic fine-tuning, we adopt distillation methods in various types, including DM [30], MTT [4], and FRePo [32] to evaluate the broad applicability of our paradigm. Dynamic memory is initialized with random noise. We use a small 3D CNN (referred to as MiniC3D) for distillation. The $\mathcal{H}$ network used for combining static and dynamic memory is also a MiniC3D. For more details, please refer to the supplementary.

Fair Comparison between the baseline and our method. We rigorously ensure that the total storage space for static memory, dynamic memory, and the $\mathcal{H}$ network is smaller than the corresponding IPC (Instance Per Class). Specif-

Dataset		MiniUCF		HMDB51	
IPC		1	5	1	5
Full Dataset		57.22 ± 0.14		28.58 ± 0.69	
Coreset Selection	Random	9.9 ± 0.8	22.9 ± 1.1	4.6 ± 0.5	6.6 ± 0.7
	Herding [26]	12.7 ± 1.6	25.8 ± 0.3	3.8 ± 0.2	8.5 ± 0.4
	K-Center [18]	11.5 ± 0.7	23.0 ± 1.3	3.1 ± 0.1	5.2 ± 0.3
Dataset Distillation	DM [30]	15.3 ± 1.1	25.7 ± 0.2	6.1 ± 0.2	8.0 ± 0.2
	MTT [4]	19.0 ± 0.1	28.4 ± 0.7	6.6 ± 0.5	8.4 ± 0.6
	FRePo [32]	20.3 ± 0.5	30.2 ± 1.7	7.2 ± 0.8	9.6 ± 0.7
	Static-DC	13.7 ± 1.1	24.7 ± 0.5	5.1 ± 0.9	7.8 ± 0.4
	DM [30]+Ours	17.5 ± 0.1	27.2 ± 0.4	6.0 ± 0.4	8.2 ± 0.1
	MTT [4]+Ours	$\mathbf{23.3 \pm 0.6}$	28.3 ± 0.0	6.5 ± 0.1	8.9 ± 0.6
	FRePo [32]+Ours	22.0 ± 1.0	$\mathbf{31.2 \pm 0.7}$	$\mathbf{8.6 \pm 0.5}$	$\mathbf{10.3 \pm 0.6}$

(a) Accuracy

Dataset	MiniUCF		HMDB51	
IPC	1	5	1	5
Full Dataset	9.81 GB		4.93 GB	
Random Herding [26] K-Center [18]	115 MB	586 MB	115 MB	586 MB
DM [30] MTT [4] FRePo [32]	115 MB	586 MB	115 MB	586 MB
Static-DC	8 MB	36 MB	8 MB	36 MB
DM [30]+Ours	94 MB	455 MB	94 MB	455 MB
MTT [4]+Ours	94 MB	455 MB	94 MB	455 MB
FRePo [32]+Ours	48 MB	228 MB	48 MB	228 MB

(b) Storage

Table 2. Results of baselines and our method on small-scale datasets. Top-1 test accuracies (%) and memory storage budget (MB or GB) are reported. Storage represents the **total size of tensors**, assuming the data is stored as floats. Our method uses no more than **42%** storage compared with the naively adapted method for FRePo, while 82% for DM and MTT. We use the storage of coreset selection methods as a reference and color-code the high, comparable, and low storage. IPC: Instance(s) Per Class.

Dataset	Kinetics-400		SSv2	
IPC	1	5	1	5
Full Dataset	34.6 ± 0.5		29.0 ± 0.6	
Random	3.0 ± 0.1	5.6 ± 0.0	3.3 ± 0.1	3.9 ± 0.1
DM [30]	6.3 ± 0.0	9.1 ± 0.9	3.6 ± 0.0	4.1 ± 0.0
MTT [4]	3.8 ± 0.2	9.1 ± 0.3	3.9 ± 0.1	6.3 ± 0.3
Static-DC	4.6 ± 0.2	6.6 ± 0.2	3.9 ± 0.1	4.1 ± 0.0
DM[30]+Ours	6.3 ± 0.2	7.0 ± 0.1	4.0 ± 0.1	3.8 ± 0.1
MTT[4]+Ours	6.3 ± 0.1	$\mathbf{11.5 \pm 0.5}$	$\mathbf{5.5 \pm 0.1}$	$\mathbf{8.3 \pm 0.2}$

Table 3. Top-5 accuracy on Kinetics-400 [3] and SSv2 [8].

ically, on DM and MTT, we use no more than 82% of the storage space corresponding to the baseline, which amounts to 2 static memory with 2 dynamic memory for every instance. On FRePo, we use no more than 42% of the storage space corresponding to the baseline, which means 1 static memory with 1 dynamic memory for every instance.

Evaluation of Distilled Dataset. Naturally, we evaluate how well our synthetic data performs on architectures used to distill it. When evaluating data distilled by FRePo, the results should be considered as a reference only due to the label learning conducted by the method itself and the use of a different optimizer. We also evaluate how well our synthetic data performs on different architectures from the one used to distill it on the MiniUCF, 1 instance per class task.

Hyper-Parameters. Considering the numerous parameters involved in the experiments, we detail the parameter settings for all experiments in the supplementary.

5.4. Results

We show the results of our small-scale experiments in Tab. 2 and large-scale in Tab. 3. The full dataset indicates the accuracy of the network trained on the full dataset.

Comparison to Coreset Method. Following image distillation, we compare our method with coreset selection methods on MiniUCF and HMDB51 in Tab. 2. In the majority of cases, our approach outperforms the coreset selection. Regarding coreset methods, we also observe: (1) On average, the herding method proves to be the most effective coreset approach. (2) With an increase in IPC, the performance of herding exhibits a notable improvement. These conclusions align with earlier experiments [31] in image distillation, lending credibility to our findings.

Comparison to Other Methods. We compare our final method with other methods we proposed in Fig. 1. Among the three methods we proposed, Static-DC (Fig. 1(b)) exhibits the poorest performance. Compared to both the coreset method and Static-DC, the naively adapted method (Fig. 1(a)) shows a significant improvement. This underscores the applicability of existing image distillation techniques to videos and the effects of dynamic information in video understanding. In comparison, our final method (corresponding to Fig. 1(c)) achieves a remarkable superiority over all other methods through static and dynamic disentangle learning. In most cases, our method could enhance the current distillation methods while requiring less storage. However, the performance of our method on Kinetics-400 is not as strong as on the other two smaller datasets. This is because Kinetics-400 itself has a much larger number of categories and samples compared to the other two smaller datasets, which increases the difficulty and cost of distillation. Since our $\mathcal{H}$ network is shared, having 400 categories sharing one $\mathcal{H}$ network in Kinetics-400 might cause interference during learning. Allocating different $\mathcal{H}$ networks to different categories or using a more complex $\mathcal{H}$ network could potentially offer better improvement in our method. However, this approach would impose a greater training burden and significantly reduce the practical value of distillation, so we do not conduct larger-scale experiments.

Cross Architecture Generalization. We also show the result of cross-architecture generalization in Tab. 4. The experimental results indicate that the data obtained by static-dynamic disentanglement performs much better on other networks compared to the naively adapted method.

	Evaluation Model		
	ConvNet3D	CNN+GRU	CNN+LSTM
Random	9.9 ± 0.8	6.2 ± 0.8	6.5 ± 0.3
DM [30]	15.3 ± 1.1	9.9 ± 0.7	9.2 ± 0.3
DM[30]+Ours	$\mathbf{17.5 \pm 0.1}$	$\mathbf{12.0 \pm 0.7}$	$\mathbf{10.3 \pm 0.2}$
MTT [4]	19.0 ± 0.1	8.4 ± 0.5	7.3 ± 0.4
MTT[4]+Ours	$\mathbf{23.3 \pm 0.6}$	$\mathbf{14.8 \pm 0.1}$	$\mathbf{13.4 \pm 0.2}$

Table 4. Cross-architecture generalization for MiniUCF IPC=1. CNN+GRU and CNN+LSTM are detailed in the supplementary.

SPC	DPC	Acc (%)	Storage	S1 Time	S1 GPU Memory
1	0	13.7 ± 1.1	8 MB	68 s/iter	4,651 MiB
	1	17.5 ± 0.5	48 MB		
	2	19.6 ± 1.2	79 MB		
	3	20.6 ± 0.2	111 MB		
2	0	20.4 ± 0.5	14 MB	156 s/iter	6,579 MiB
	1	22.3 ± 0.0	54 MB		
	2	23.3 ± 0.6	94 MB		
	3	-	117 MB		

Table 5. Results and stage 1 cost for MiniUCF IPC=1 with different ratios of static and dynamic memory. SPC: static memory per class. DPC: dynamic memory per class. For MiniUCF IPC=1, the storage for naive methods is 115 MB. We ensure that the storage used by our method is less than 115 MB. S1 Time: The time required for each iteration (s/iter) at Stage 1 (static learning stage). S1 GPU Memory: The GPU memory (MiB) required at Stage 1.

5.5. Ablation Study

Ratios of Static and Dynamic. We report results for MiniUCF 1 Instance per class task with different static and dynamic memory ratios in Tab. 5. We can find that increasing the quantity of static and dynamic memory both improves the scores. However, as their numbers increase, the time and GPU memory required for Static Learning and the training convergence time for Dynamic Fine-tuning under the same computational power will also increase (Tab. 5). Considering training efficiency and effectiveness while ensuring the storage does not exceed the corresponding IPC, a balanced choice is to use 2 static with 2 dynamics for every instance.

Impact of Video Dynamics on Distillation. Actions in the video exhibit varying degrees of dynamics. To explore the impact of video dynamics, we categorized all classes of MiniUCF into two groups (relatively static, highly dynamic) based on their level of dynamics, calculated by the average Hamming distance between inter-frame features for each class, and compared their test accuracy on networks trained with distilled data (Fig. 8). We observe that (1) data distilled by the Static-DC method is more sensitive for static classes, which aligns with our expectations as this method generates data that lacks dynamic information, akin to a "boring video". (2) The naively adapted MTT, in comparison, can distill more useful information but still shows higher scores for static classes than dynamic classes. (3) MTT+Ours, however, demonstrates better distillation of dynamic information compared to the previous methods and exhibits significant improvements on dynamic classes.

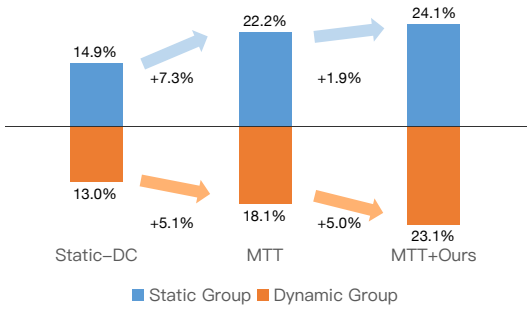


Figure 8. Test accuracies of static and dynamic group on network trained with distilled data.

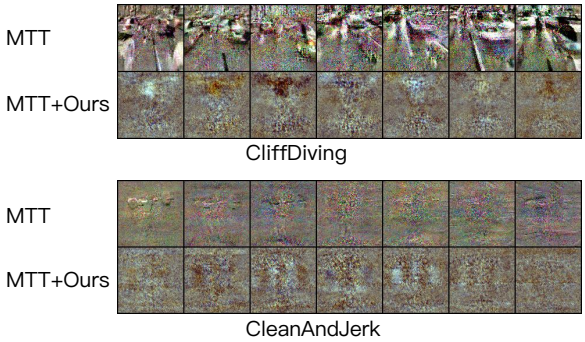


Figure 9. Visualized inter-frame differences of videos distilled by MTT and MTT+Ours for MiniUCF IPC=1.

5.6. Visualization

To observe the temporal changes in the distilled videos, we sampled frames from the videos obtained using different methods and visualized their *inter-frame differences*. We show two examples in Fig. 9 and more in the supplementary. Although visually abstract, we can still conclude that the distilled videos indeed exhibit temporal variations.

6. Conclusion

In this work, we provide the first systematic study of video distillation. We propose a taxonomy to categorize methods based on key factors including the number of frames and segment length. With extensive experiments, we revealed that more frames provide marginal gains at greatly increased costs. Then, we proposed our method that disentangles static and dynamic information and achieves SOTA with efficient storage. We believe our paradigm will pave a new way for video distillation.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China under Grants 62306175, National Key Research and Development Project of China (No.2022ZD0160102, No.2021ZD0110704), Shanghai Artificial Intelligence Laboratory, XPLORER PRIZE grants.

References

[1] Simon Baker, Daniel Scharstein, JP Lewis, Stefan Roth, Michael J Black, and Richard Szeliski. A database and evaluation methodology for optical flow. *IJCV*, 92:1–31, 2011. 2

[2] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, page 4, 2021. 3, 4

[3] João Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 4724–4733, 2017. 1, 2, 6, 7, 4

[4] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *CVPR*, 2022. 1, 2, 6, 7, 8

[5] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014. 4

[6] Justin Cui, Ruochen Wang, Si Si, and Cho-Jui Hsieh. Scaling up dataset distillation to imagenet-1k with constant memory. In *ICML*, pages 6565–6590, 2023. 1, 2

[7] Zhiwei Deng and Olga Russakovsky. Remember the past: Distilling datasets into addressable memories for neural networks. In *NeurIPS*, 2022. 1, 2

[8] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzyńska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense, 2017. 6, 7, 4

[9] De-An Huang, Vignesh Ramanathan, Dhruv Mahajan, Lorenzo Torresani, Manohar Paluri, Li Fei-Fei, and Juan Carlos Niebles. What makes a video a video: Analyzing temporal information in video understanding models and datasets. In *CVPR*, pages 7366–7375, 2018. 3

[10] Jang-Hyun Kim, Jinuk Kim, Seong Joon Oh, Sangdoo Yun, Hwanjun Song, Joonhyun Jeong, Jung-Woo Ha, and Hyun Oh Song. Dataset condensation via efficient synthetic-data parameterization. In *ICML*, 2022. 1, 2

[11] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: A large video database for human motion recognition. In *ICCV*, pages 2556–2563, 2011. 1, 6, 2, 4

[12] Yongqi Li and Wenjie Li. Data distillation for text classification. *arXiv preprint arXiv:2104.08448*, 2021. 1

[13] Songhua Liu, Kai Wang, Xingyi Yang, Jingwen Ye, and Xinchao Wang. Dataset distillation via factorization. In *NeurIPS*, 2022. 1, 2

[14] Xin Liu, Silvia L. Pintea, Fatemeh Karimi Nejadasl, Olaf Booij, and Jan C. van Gemert. No frame left behind: Full video action recognition. In *CVPR*, pages 14892–14901, 2021. 3

[15] Noel Loo, Ramin Hasani, Alexander Amini, and Daniela Rus. Efficient dataset distillation using random feature approximation. In *NeurIPS*, 2022. 1, 2

[16] Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. *arXiv preprint arXiv:2011.00050*, 2020. 2

[17] Timothy Nguyen, Roman Novak, Lechao Xiao, and Jaehoon Lee. Dataset distillation with infinitely wide convolutional networks. In *NeurIPS*, 2021. 1, 2

[18] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *ICLR*, 2018. 6, 7

[19] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *CoRR*, abs/1212.0402, 2012. 1, 6, 2

[20] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *NeurIPS*, pages 10078–10093. Curran Associates, Inc., 2022. 3

[21] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, pages 4489–4497, 2015. 2, 6

[22] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, pages 6450–6459, 2018. 2

[23] Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features. In *CVPR*, 2022. 1, 2

[24] Mengmeng Wang, Jiazheng Xing, and Yong Liu. Actionclip: A new paradigm for video action recognition. *CoRR*, abs/2109.08472, 2021. 3

[25] Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018. 1, 2, 6

[26] Max Welling. Herding dynamical weights to learn. In *ICML*, 2009. 6, 7

[27] Xindi Wu, Zhiwei Deng, and Olga Russakovsky. Multimodal dataset distillation for image-text retrieval. *arXiv preprint arXiv:2308.07545*, 2023. 1

[28] Zhe Xu, Yuzhong Chen, Menghai Pan, Huiyuan Chen, Mahashweta Das, Hao Yang, and Hanghang Tong. Kernel ridge regression-based graph dataset distillation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2850–2861, 2023. 1

[29] Bo Zhao and Hakan Bilen. Dataset condensation with differentiable siamese augmentation. In *ICML*, 2021. 1, 2

[30] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *WACV*, 2023. 1, 2, 4, 5, 6, 7, 8, 3

[31] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020. 1, 2, 4, 6, 7

[32] Yongchao Zhou, Ehsan Nezhadarya, and Jimmy Ba. Dataset distillation using neural feature regression. *arXiv preprint arXiv:2206.00719*, 2022. 1, 2, 6, 7, 3